1 Forelesning 7

1.1 Problem vi skal løse idag

1.2 Priority Queue

- En datastruktur med følgende operasjoner
  - `add(T element)`
  - `T findMin()` (eller `findMax`)
  - `T removeMin()` (eller `removeMax`)

1.2.1 Linked list

- `add(T element)` - $O(1)$
- `T findMin()` - $O(n)$
- `T removeMin()` - $O(n)$

1.2.2 Sortert Liste

Data-invariant: Listen er sortert

- `add(T element)` - $O(n)$
- `T peekMin()` - $O(1)$
- `T removeMin()` - $O(1)$

1.3 Eksempel på bruk av priority queue

- Kø på legevakten
  - De som er mest syk kommer til først
- Operativsystem
  - Viktige prosesser får høy prioritet
- Graf algoritmer - mer i kaptitel 4
- Huffman data compression
  - De vanligste sekvensene får kort kode
  - De minst vanlige får lang kode

1.4 Binary Tree - Datainvariant

- 1 root
- Hver node har opp til 2 children
- Hvis a er child av b så sier vi at b er parent av a
- Hver node har 1 parent untatt root som har 0.

- De nederste nodene kalles *leaf*

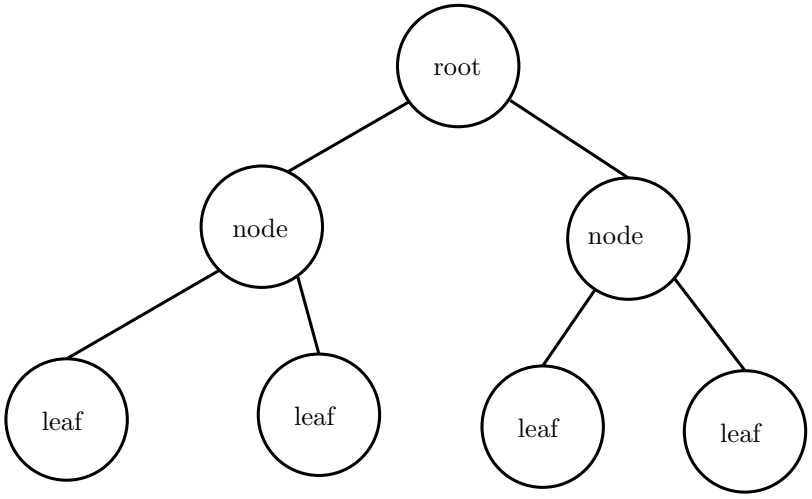

Figure 1: Binært tre

1.5 Complete binary tree

Et komplett binært tre består av halvparten som løv, og den andre halvparten av noder og røtter. Høyden til et komplett binært tre er $\log(n)$.

1.6 Heap - data invariant

- Er et binary tree med følgende krav:
  - Hver node er mindre enn alle sin children
  - Treet er balansert

1.6.1 Kjøretid

- `add(T element)` - $O(\log(n))$
- `T peekMin` - $O(1)$
- `T removeMin()` - $O(\log(n))$

1.7 Heap Sort

- Kan vi bruke Heap til å sortere?
- La oss implementere HeapSort.

1.8 Median Data Structure

Use sorted ArrayList

- `add()` - $O(n)$
- `findMedian()` - $O(1)$

Bruk to heap datastrukturer, en larger alle elementer midnre enn median, den andre lagrer alle elementer større enn median.

- `add()` - $O(\log n)$
- `findMedian()` - $O(1)$

1.9 Heap - Initialize

Gitt en liste, hvor for kan vi legge inn alle elementene i en Heap?

```
for (T elem : list)
    heap.add(elem)
```

Dette gir kjøretid $O(n \log(n))$

1.9.1 Kjøretid

- Nederste level: 0
- Nest nederste: $\frac{n}{2} \cdot 1$
- Nest nest nederste: $\frac{n}{4} \cdot 2$

Totalt:

$$\sum_{i=1}^{\log(n)} \frac{n \cdot i}{2^i} \in O(n)$$